Session d'exercices – Pointeurs
Moyenne glissante

En traitement du signal, on sait qu'un signal échantilloné a une certaine fréquence d'échantillonage peut être parfaitement reconstruit si la fréquence du signal est strictement inférieure à la moitié de la fréquence d'échantillonage. On appelle cette fréquence maximale la fréquence de Nyquist. Dans cet exercice, nous définirons deux structures représentant les caractéristiques d'un signal sinusoïdal simple et d'un échantillonage et utiliserons ensuite ces structures pour vérifier la théorie en échantillonant des signaux de fréquences diverses. A noter que dans cet exercice, un signal complet est défini comme étant une somme de signaux sinusoïdaux.

Si la théorie sous-jacente n'est plus très claire pour vous, vous trouverez des informations sur les pages suivantes :

1. Signaux sinusoïdes
2. Echantillonage :
3. Théorème d'échantillonage :

a) [Difficulté : *] Ecrivez la structure

`Sinusoide`,

qui doit contenir 3 champs représentant les 3 constantes inhérentes d'un signal sinusoïdal pur : la fréquence, l'amplitude et le déphasage.

b) [Difficulté : *] Écrivez la structure

`Echantillonage`,

qui doit contenir 2 champs représentant l'instant auquel l'échantillonage commence et sa fréquence.

Continuez en écrivant tout ce qui manque pour compléter le programme, incluant ce qui suit :

c) [Difficulté : **] Écrivez la fonction

`double movAvg(float *in, float *out, int m)`,

qui prend en argument un réel et une structure Sinusoïde et retourne la valeur de la sinusoïde à un instant t. Si l'utilisation des 3 constantes d'une sinusoïde n'est pas claire, se référer aux pages de théorie plus haut.

d) [Difficulté : **] Écrivez la fonction

`void movAvg(float *in, float *out, int m)`,

qui prend comme arguments les adresses de début du tableau de données et du tableau contenant les moyennes, ainsi que la taille du groupe d'éléments utilisés pour calculer chaque moyenne. Cette fonction remplit le tableau des moyennes suivant le premier exemple (simple moving average) et affiche tous les éléments de ce tableau comme dans l'exemple de sortie au bas de cette page.

e) [Difficulté : ***] Écrivez la fonction

`void movAvgFull(float *in, float *out, int m)`,

qui prend comme arguments les adresses de début du tableau de données et du tableau contenant les moyennes, ainsi que la taille du groupe d'éléments utilisés pour calculer chaque moyenne. Cette fonction remplit le tableau des moyennes suivant le deuxième exemple (full moving average) et affiche tous les éléments de ce tableau comme dans l'exemple de sortie au bas de cette page.

Continuez en écrivant tout ce qui manque pour compléter le programme, incluant ce qui suit :

f) [Difficulté : *] Déclarez et initialisez un tableau de données de 10 éléments en virgule flottante dont les valeurs sont : 1, 2, 3, 4, 5, 5, 4, 3, 2 et 1.

g) [Difficulté : *] Votre programme doit être appelé avec un paramètre. Ce paramètre est le nombre d'éléments utilisés pour calculer les moyennes. (hint : il faut utiliser `argc/argv`, et pas `scanf` !)
   — Si ce paramètre n'est pas spécifié ou plusieurs paramètres sont spécifiés, le programme doit afficher le message

   Incorrect number of program arguments.

   et ensuite terminer.

— Si la valeur de ce paramètre est plus petite que 1 ou plus grande que la taille du tableau de données, le programme doit afficher le message

Invalid program argument value.

et ensuite terminer.

h) [Difficulté : *] Le programme doit faire appel aux fonctions `movAvg` et `movAvgFull`.

Exemple de sortie du programme pour `m = 3` :